mdn web docs

HTTP Client hints

Client hints are a set of HTTP request header fields that a server can proactively request from a client to get information about the device, network, user, and user-agent-specific preferences. The server can determine which resources to send, based on the information that the client chooses to provide.

The set of "hint" headers are listed in the topic HTTP Headers and summarized below.

Overview

A server must announce that it supports client hints, using the `Accept-CH` header to specify the hints that it is interested in receiving. When a client that supports client hints receives the `Accept-CH` header it can choose to append some or all of the listed client hint headers in its subsequent requests.

For example, following `Accept-CH` in a response below, the client could append `Width`, `Downlink` and `Sec-CH-UA` headers to all subsequent requests.

HTTP

```
Accept-CH: Width, Downlink, Sec-CH-UA
```

This approach is efficient in that the server only requests the information that it is able to usefully handle. It is also relatively "privacy-preserving", in that it is up to the client to decide what information it can safely share.

There is a small set of low entropy client hint headers that may be sent by a client even if not requested.

> **Note:** Client hints can also be specified in HTML using the `<meta>` element with the `http-equiv` attribute.
>
> HTML
>
> ```
> <meta http-equiv="Accept-CH" content="Width, Downlink, Sec-CH-UA" />
> ```

Caching and Client Hints

Client hints that determine which resources are sent in responses should generally also be included in the affected response's `Vary` header. This ensures that a different resource is cached for every different value of the hint header.

HTTP

```
Vary: Accept, Width, ECT
```

You may prefer to omit specifying `Vary` or use some other strategy for client hint headers where the value changes a lot, as this effectively makes the resource uncacheable. (A new cache entry is created for every unique value.) This applies in particular to network client hints like `Downlink` and `RTT`. For more information see HTTP Caching > Vary.

Hint life-time

A server specifies the client hint headers that it is interested in getting in the `Accept-CH` response header. The user agent appends the requested client hint headers, or at least the subset that it wants to share with that server, to all subsequent requests in the current browsing session.

In other words, the request for a specific set of hints does not expire until the browser is shut down.

A server can replace the set of client hints it is interested in receiving by resending the `Accept-CH` response header with a new list. For example, to stop requesting any hints it would send `Accept-CH` with an empty list.

> **Note:** The client hints set for a particular origin can also be cleared by sending a `Clear-Site-Data: "clientHints"` response header for a URL inside that origin.

Low entropy hints

Client hints are broadly divided into high and low entropy hints. The low entropy hints are those that don't give away much information that might be used to create a fingerprinting for a user. They may be sent by default on every client request, irrespective of the server `Accept-CH` response header, depending on the permission policy. Low entropy hints are:

- `Save-Data` ,
- `Sec-CH-UA` ,
- `Sec-CH-UA-Mobile` , and
- `Sec-CH-UA-Platform` .

High entropy hints

The high entropy hints are those that have the potential to give away more information that can be used for user fingerprinting, and therefore are gated in such a way that the user agent can make a decision whether to provide them. The decision might be based on user preferences, a permission request, or the permission policy. All client hints that are not low entropy hints are considered high entropy hints.

Critical client hints

A *critical client hint* is one where applying the response may significantly change the rendered page, potentially in a way that is jarring or will affect usability, and therefore which must be applied before the content is rendered. For example, `Sec-CH-Prefers-Reduced-Motion` is commonly treated as a critical hint, because it might markedly affect the behavior of animations, and because a user who chooses this preference needs it to be set.

A server can use the `Critical-CH` response header along with `Accept-CH` to specify that an accepted client hint is also a critical client hint (a header in `Critical-CH` must also appear in `Accept-CH`). User agents receiving a response with `Critical-CH` must check if the indicated critical headers were sent in the original request. If not, then the user agent will retry the request rather than render the page. This approach ensures that client preferences set using critical client hints are always used, even if not included in the first request, or if the server configuration changes.

For example, in this case, the server tells a client via `Accept-CH` that it accepts `Sec-CH-Prefers-Reduced-Motion`, and `Critical-CH` is used to specify that `Sec-CH-Prefers-Reduced-Motion` is considered a critical client hint:

HTTP

```
HTTP/1.1 200 OK
Content-Type: text/html
Accept-CH: Sec-CH-Prefers-Reduced-Motion
Vary: Sec-CH-Prefers-Reduced-Motion
Critical-CH: Sec-CH-Prefers-Reduced-Motion
```

> **Note:** We've also specified `Sec-CH-Prefers-Reduced-Motion` in the `Vary` header to indicate to the browser that the served content will differ based on this header value, even if the URL stays the same, so the browser shouldn't just use an existing cached response and instead should cache this response separately. Each header listed in the `Critical-CH` header should also be present in the `Accept-CH` and `Vary` headers.

As `Sec-CH-Prefers-Reduced-Motion` is a critical hint that was not in the original request, the client automatically retries the request — this time telling the server via `Sec-CH-Prefers-Reduced-Motion` that it has a user preference for reduced-motion animations.

HTTP

```
GET / HTTP/1.1
Host: example.com
Sec-CH-Prefers-Reduced-Motion: "reduce"
```

Hint types

User agent client hints

User agent (UA) client hint headers allow a server to vary responses based on the user agent (browser), operating system, and device. For a list of `Sec-CH-UA-*` headers, see User agent client hints headers.

Client hints are available to web page JavaScript via the User Agent Client Hints API.

> **Note:** Servers currently get most of the same information by parsing the `User-Agent` header. For historical reasons this header contains a lot of largely irrelevant information, and information that might be used to identify a *particular user*. UA client hints provide a more efficient and privacy preserving way of getting the desired information. They are eventually expected to replace this older approach.

> **Note:** User-agent client hints are not available inside fenced frames because they rely on permissions policy delegation, which could be used to leak data.

User preference media features client hints

User Preference Media Features Client Hints allow a server to vary responses based on a user agent's preferences for CSS media features such as color scheme or reduced motion. Headers include: `Sec-CH-Prefers-Reduced-Motion`, `Sec-CH-Prefers-Color-Scheme`.

Device client hints

Device client hints allow a server to vary responses based on device characteristics including available memory and screen properties. Headers include: `Device-Memory`, `Width`, `Viewport-Width`.

Network client hints

Network client hints allow a server to vary responses based on the user's choice, network bandwidth, and latency. Headers include: `Save-Data`, `Downlink`, `ECT`, `RTT`.

See also

- Client Hints headers
- `Vary` HTTP Header
- Client Hints Infrastructure
- User Agent Client Hints API
- Improving user privacy and developer experience with User-Agent Client Hints (developer.chrome.com)